computers, dedicated processors and/or dedicated hard wired logic may be used to construct alternative equivalent embodiments of the present invention.

Those skilled in the art will appreciate that the program steps and associated data used to implement the embodiments described above can be implemented using disc storage as well as other forms of storage such as for example Read Only Memory (ROM) devices, Random Access Memory (RAM) devices; optical storage elements, magnetic storage elements, magneto-optical storage elements, flash memory, core memory and/or other equivalent storage technologies without departing from the present invention. Such alternative storage devices should be considered equivalents.

The present invention, as described in embodiments herein, is implemented using a programmed processor executing programming instructions that are broadly described above in flow chart form that can be stored on any suitable electronic storage medium or transmitted over any suitable electronic communication medium. However, those skilled in the art will appreciate that the processes described above can be implemented in any number of variations and in many suitable programming languages without departing from the present invention. For example, the order of certain operations carried out can often be varied, additional operations can be added or operations can be deleted without departing from the invention. Error trapping can be added and/or enhanced and variations can be made in user interface and information presentation without departing from the present invention. Such variations are contemplated and considered equivalent.

While the invention has been described in conjunction with specific embodiments, it is evident that many alternatives, modifications, permutations and variations will become apparent to those skilled in the art in light of the foregoing description. Accordingly, it is intended that the present invention embrace all such alternatives, modifications and variations as fall within the scope of the appended claims.

What is claimed is:

1. An in-circuit emulation system, comprising:

a microcontroller;

a virtual microcontroller coupled to and executing instructions in lock-step with the microcontroller, and wherein the microcontroller sends I/O read data to the virtual microcontroller;

the virtual microcontroller having means for detecting a sequence of instructions comprising an I/O read instruction followed by a conditional jump instruction, and for computing a conditional jump address prior to receipt of I/O read data from the microcontroller; and

the virtual microcontroller further having means for determining after receipt of the I/O read data from the microcontroller whether to proceed with instruction execution at a next consecutive address or at the conditional jump address.

2. The apparatus according to claim 1, wherein the conditional jump address is computed while the I/O read data are sent from the microcontroller to the virtual microcontroller.

3. The apparatus according to claim 1, wherein the microcontroller sets a zero flag if an I/O read test condition is met.

4. The apparatus according to claim 3, wherein the jump condition is met if the zero flag is set.

5. The apparatus according to claim 1, wherein the virtual microcontroller is implemented in a Field Programmable Gate Array.

6. In an in-circuit emulation system having a microcontroller coupled to and operating in lock-step with a virtual microcontroller, a method of handling conditional jumps in the virtual microcontroller, comprising:

detecting a sequence of instructions comprising an I/O read instruction followed by a conditional jump instruction;

computing a conditional jump address prior to receipt of I/O read data from the microcontroller; and

determining after receipt of the I/O read data from the microcontroller whether to proceed with instruction execution at a next consecutive address or at the conditional jump address.

7. The method according to claim 6, further comprising executing a next consecutive instruction in the event a conditional jump condition is not met.

8. The method according to claim 6, further comprising executing an instruction at the conditional jump address in the event the conditional jump condition is met.

9. The method according to claim 6, wherein the conditional jump address is computed while the I/O read data are sent from the microcontroller to the virtual microcontroller.

10. The method according to claim 6, wherein the microcontroller sets a zero flag if an I/O read test condition is met.

11. The method according to claim 10, wherein the jump condition is met if the zero flag is set.

12. The method according to claim 6, wherein the virtual microcontroller is implemented in a Field Programmable Gate Array.

13. The method according to claim 6, stored as instructions stored in an electronic storage medium for execution as program steps on a programmed processor forming a part of the virtual microcontroller.

14. In an in-circuit emulation system having a device under test coupled to and operating in lock-step with a virtual processor, a method of handling conditional jumps in the virtual processor, comprising:

detecting a sequence of instructions comprising an I/O read instruction followed by a conditional jump instruction;

computing a conditional jump address prior to receipt of I/O read data from the virtual processor; and

determining after receipt of the I/O read data from the device under test whether to proceed with instruction execution at a next consecutive address or at the conditional jump address.

15. The method according to claim 14, further comprising executing a next consecutive instruction in the event a conditional jump condition is not met.

16. The method according to claim 14, further comprising executing an instruction at the conditional jump address in the event the conditional jump condition is met.

17. The method according to claim 14, wherein the conditional jump address is computed while the I/O read data are sent from the device under test to the virtual processor.

18. The method according to claim 14, wherein the device under test sets a zero flag if an I/O read test condition is met.

19. The method according to claim 18, wherein the jump condition is met if the zero flag is set.

20. The method according to claim 14, wherein the virtual processor is implemented in a Field Programmable Gate Array.